Server-based multi-standard home network bridging

Publication number: CN1398469 (A)
Publication date: 2003-02-19
Inventor(s): MOONEN J R [NL]; SHTEYN Y E [NL] +
Applicant(s): KONINKL PHILIPS ELECTRONICS NV [NL] +
Classification:
- international: ***H04L12/28; H04L12/46;*** **H04L12/28; H04L12/46;** (IPC1-7): H04L12/12; H04L12/46; H04L12/64
- European: H04L12/28H; H04L12/28H1; H04L12/28H5A; H04L12/46B7
Application number: CN20018002165 20010720
Priority number(s): US20000616632 20000726

Also published as:
WO0209350 (A2)
WO0209350 (A3)
JP2004505499 (T)
EP1307998 (A2)

Abstract not available for CN 1398469 (A)
Abstract of corresponding document: **WO 0209350 (A2)**

A bridge in a home network couples first and second clusters of devices. The clusters have different software architectures. The bridge is connected to a server on the Internet. This server offers a lookup service for some set of standards, and allows a bridge to locate and download the appropriate translation modules for allowing a device in the first cluster to interact with the second cluster.

[19] 中华人民共和国国家知识产权局

[51] Int. Cl[7]
H04L 12/12
H04L 12/64 H04L 12/46

[12] 发明专利申请公开说明书

[21] 申请号 01802165.4

[43] 公开日 2003 年 2 月 19 日

[11] 公开号 CN 1398469A

[22] 申请日 2001.7.20 [21] 申请号 01802165.4
[30] 优先权
[32] 2000. 7.26 [33] US [31] 09/616632
[86] 国际申请 PCT/EP01/08552 2001.7.20
[87] 国际公布 WO02/09350 英 2002.1.31
[85] 进入国家阶段日期 2002.3.25
[71] 申请人 皇家菲利浦电子有限公司
地址 荷兰艾恩德霍芬
[72] 发明人 J·R·穆宁 Y·E·施泰恩

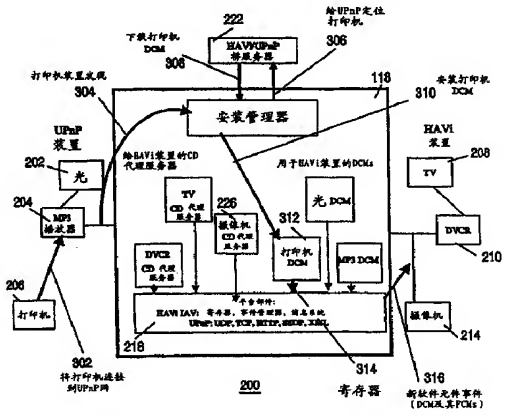
[74] 专利代理机构 中国专利代理(香港)有限公司
代理人 邹光新 陈 霁

权利要求书 1 页 说明书 9 页 附图 3 页

[54] 发明名称 基于服务器的多标准家庭网桥接方法

[57] 摘要

连接第一族和第二族装置的家庭网中的一种桥。这些族具有不同的软件体系结构。这种桥跟因特网上的服务器连接。这个服务器为某些组标准提供查阅服务，并且允许桥寻找和下载适当的翻译模块，从而使第一族里的装置能够跟第二族相互作用。



ISSN1008-4274

1. 一种提供服务给家庭网（100）用户的方法，其中：

- 该方法包括让网络中第一族（110）的部件（116）能够跟家庭网的第二族（102）相互作用；

- 第一族具有第一个软件体系结构；

- 第二族具有不同于第一个体系结构的第二个软件体系结构；

- 第一族和第二族通过桥（118）连接；

- 该方法包括：

- 让这一族外部的服务器（124）接收第一族中具有第一个软件表示的一个部件的引用；和

- 给桥提供跟引用有关的一个翻译模块（128），在桥上安装这个模块的时候，用于至少部分地代表第二族中的部件。

2. 权利要求 1 的方法，其中的服务器从桥接收引用。

3. 权利要求 1 的方法，其中的桥通过因特网跟服务器联系。

4. 权利要求 1 的方法，其中的第一族包括一个 HAVi 族。

5. 权利要求 1 的方法，其中的第一族包括一个 UPnP 族。

6. 一种数据库，包括至少一个翻译模块（128），通过因特网将这个模块下载到连接第一族和第二族的桥（118）上以后，用于让第一个软件体系结构的第一个家庭网族（110）上的部件（116）能够跟第二种软件体系结构的第二个家庭网族（102）相互作用。

基于服务器的多标准家庭网桥接方法

本申请是 Yevgeniy Eugene Shteyn 于 99 年 6 月 25 日提交的第 09/340272 号美国专利申请（代理人档案 PHA 23634）的部分继续申请，这个美国申请的标题是“多个家庭网软件体系结构的桥接”。

本申请涉及在不同软件体系结构的基础之上多个网络的桥接方法。具体而言，本发明涉及家庭联网方法。

家里每个装置都采用单独一个通用的联网标准似乎可能性很小。软件体系结构的多个标准会共存，同时会出现新标准。会为这些新类型装置开发出新标准接口，满足这些标准的要求。家庭应用是为了充分地利用家里的所有装置，但是不能符合正在使用或者将来要使用的所有联网标准。同样，这些装置自己也不能支持所有家庭联网标准。因此，在不同子网或者族之间需要桥接器，每一个都对应一个标准。桥用来以透明方式代表装置，符合第一种网络族的第一种标准，作为符合第二种网络族中第二种标准的装置。结果是家庭网中为第一种标准编写的软件应用程序和为第二种标准编写的软件应用程序都统一了起来。

一般情况下，网络里的装置是利用跟装置接口兼容的一组消息来控制的。装置和软件应用程序之间的协同工作能力取决于唯一标识的标准接口。一旦应用程序知道了装置的唯一标识，它就知道这个装置的接口，它就能够通过向它发送消息来控制这个装置。对于应用程序而言，它是直接将标准消息发送给装置本身，还是间接地通过一个软件将这些消息翻译成不同的一组消息，（最终）在受控装置那里得到相同的效果或者状态改变，没有任何差别。因此，可以将采用多个不同标准的网络之间的桥看作一个多标准装置。也就是说，这个桥符合所有这些标准，用软件将第一个标准跟第二个标准衔接起来。例如，在符合标准 A 的 5 个装置，符合不同于标准 A 的标准 B 的 3 个装置之间采用一个桥。这个桥有 3 个翻译模块用于从 A 翻译到 B，有 5 个模块用于从 B 翻译到 A。因此，只能够应用标准 A 的软件应用程序就能够控制所有 8 个装置。

Yevgeniy Eugene Shteyn 于 99 年 6 月 25 日提交，美国序列号是

09/340272（代理人档案号 PHA 23634）的“多个家庭网软件体系结构的桥接”涉及将不同软件体系结构的家庭网桥接起来，在这里将它引入作为参考。对第一个网络中装置和服务的软件表示的引用是自动地产生的。这些引用从语义上讲足以自动产生至少部分功能跟第二种网络等价的软件表示，从而能够从第二种网络访问第一种网络的装置和服务。这篇文献还提到了 HAVi、家庭 API 和 Jini 软件体系结构。

下面，“翻译模块”这个术语包括“软件表示”的概念，也就是物理装置或者网络服务或者它的子集的软件表示，从而使有关的消息传递或者控制软件能够访问这个装置或者服务。

桥最好具有以下功能：检测被桥接的网络中是否增加了装置，识别增加的装置的类型，如果这个装置有可能对其它网络感兴趣就为识别了类型的装置寻找翻译模块，按照这个网络使用的标准需要的程序在其它网络里安装翻译模块。

成功的标准会继续为跟这些标准有关的领域内新开发的装置定义标准接口。这就需要开发伴随的翻译模块，从而能够在不同标准的网络内代表这些新装置。结果，桥不能够为将来的所有相关装置包括所有的嵌入翻译模块。

所以，本发明提出了一种解决方案，其中的桥跟服务器连接，例如跟因特网连接。这个服务器为一组标准提供查阅服务，并且允许桥为家庭网寻找和下载适当的翻译模块。

具体而言，本发明涉及一种方法，用于提供服务给家庭网的用户。该方法让这个网络中的第一族的部件跟家庭网中第二族相互作用。第一族有第一个软件体系结构，第二族有不同于第一个的第二个软件体系结构。第一族和第二族之间通过桥连接。该方法让这些族外面的服务器，例如因特网上的服务器，接收第一族中具有第一个软件表示的一个部件的一个引用。该方法还为桥提供跟这个引用有关的一个翻译模块，在这个桥里安装了这个模块的时候至少部分地代表第二族里的部件。

于是服务提供商就能够为家庭网里使用的所有标准维护和更新一个翻译模块数据库。这样划分功能具有如下优点。

本发明使得桥相当“轻便”或者成本相当低，因为它不需要为家里可能连接的所有标准的所有可能装置嵌入翻译模块。只有实际桥接

便提供桥给另外一族。例如，HAVi 机顶盒有软件部件将 HAVi 族跟例如家庭网上的 UPnP 族桥接。同样，控制 UPnP 族的 PC 能够拥有软件部件，将家庭网的 UPnP 族跟 HAVi 族桥接。

下面将参考附图，通过实例更加详细地描述本发明，在这些附图中：

图 1 是一个框图，说明本发明中在两个网络之间进行桥接的原理；

图 2 说明如何将 HAVi 跟 UPnP 桥接；和

图 3 说明如何将 UPnP 跟 HAVi 桥接。

在所有附图中，相同的引用数字表示相同或者对应的功能。

如上所述，本发明的一个方面涉及如何将桥跟例如因特网上的服务器连接。这个服务器为一组标准提供查阅服务，并且允许桥查找和下载适当的翻译模块，放到家庭网里去，使第一种体系结构的子网内的装置跟第二种体系结构的子网内的装置协同工作。

图 1 是家庭网系统 100 的一个示意图，它有第一族 102 装置 104、106 和 108，符合第一个软件体系结构标准，以后叫做标准 A。系统 100 包括第二族 110 装置 112、114、116，它们符合第二个软件体系结构标准，以后叫做标准 B。族 102 和 110 通过桥 118 互连。为了在标准 A 的族 102 和标准 B 的族 110 之间进行有意义的网络相互作用，采用了翻译模块。这些模块需要参加族 102 和 110。这些模块通常都需要族的本地部件，比方说低级通信软件，才能参加这些族。不是让每个翻译模块有它自己的通信软件，将这个软件作为桥 118 的平台部件 120 的一个单元更加有效。

下面利用要将 B 装置 116 添加到系统 100 中去的一个实例来说明本发明的过程。

第一步是将 B 装置 116 跟 B 族 110 进行物理连接，也就是“引导”B 装置 116。

下一步，桥 118 将 B 装置 116 当作新增加的装置，或者是因为桥 118 周期性地扫描 B 族 110，或者它的注册/目录/查阅服务（图中没有画出），或者是因为 B 族 110 主动地通知桥 118。桥 118 有一个软件部件 122，叫做安装管理程序，它负责安装将 B 装置 116 集成到系统 100 中需要的软件部件。关于这一点在例如美国序列号 09/340272（律师文号 PHA 23634）中进行了讨论。在后一篇文献中，叫做引用工厂的

的装置需要存储和计算能力（也就是说不需要为只是潜在的桥接装置准备存储能力）（例如将新装置跟网络连接的时候的“刚好及时”桥接）。

此外，本发明使家庭网整体具有扩充能力，不会过时。由于不断发明出新装置，因此对它们的描述成为各种标准规范的一部分，这些描述被例如装置制造商或者第三方翻译并且载入桥服务器，使它们能够被用于已有的家庭网进行桥接。这一过程不需要家庭网本身具有任何内容更新机制。

另一个好处是桥服务器运营商能够获得每个用户家庭网的配置信息。这些信息能够用来帮助用户和制造商以及服务提供商。例如，见 Adrian Turner 等等于 1998 年 9 月 25 日提交的美国序列号 09/160490（代理人文档号 PHA 23500）“根据用户简档定制更新具有因特网能力的装置”，在这里将它引入作为参考。这篇文献涉及到一种服务器系统，它为具有网络能力的消费电子设备的特定终端用户保存一个用户简档，并且为这种设备，例如家庭网，的新技术功能保存一个数据库。如果用户简档跟新技术功能相同，并且用户明确表示愿意接收更新信息，用户就会通过网络得到通知能够在将来进行选择。又例如，Yevgeniy Shteyn 于 1998 年 11 月 10 日提交的美国序列号 09/189535（代理人文档号 PHA 23527）“家庭网协作的升级”，在这里将它引入作为参考。这篇文献涉及一种系统，它有一个服务器，能够访问用户家庭网上装置和能力的一个清单。这个清单是例如 HAVi、Jini 和家庭 API 体系结构提供的查阅服务。这个服务器还能够访问有关网络未来的信息的数据库。服务器确定根据这个清单和用户简档来判断用户网络中的设备的协作是否能够加强。如果有跟协作有关的功能，在这些判据的基础之上，用户就能够得到通知。从这个意义上讲，美国序列号 09/189535 涉及一种“应用建议者”概念。

本发明的另一个优点是服务器运营商能够确定市场对特定装置跟某个标准的桥接的需求。装置制造商或者另一个相关的第三方能够知道有需求出现。在服务器上获得新翻译模块的时候，就能够通知过去发出了翻译模块请求而服务器不能够提供的那些桥，说明现在能够进行升级。

注意，桥可以作为家庭网特定族里一个装置的一个软件部件，以

一个软件部件能够从注册装置的软件表示中提取信息。根据这个有关软件体系结构的方法，这个引用工厂能够查询服务目录，或者能够得知新的软件表示。同样，安装管理程序 122 接收或者提取新增加 B 装置 116 的有关信息。通过因特网 126 发送给桥服务器 124 之前描述新信息有可能被重新格式化。另外，桥 118 最好是提供家庭网 100 本地执行环境的有关信息。这些信息跟服务器 124 下载到桥 118 里的软件部件有关。关于环境的信息涉及到软件体系结构，在这种情况下是 A 标准族 102 和 B 标准族 110。这些信息也可以涉及桥 118 里的可用存储器、所用操作系统的类型、存在的虚拟机、平台库等等。在这些信息的基础之上，桥 124 能够选择最适合于系统 100 的网络环境的适当的翻译模块。

收到描述和环境信息的时候，服务器 124 利用查阅服务，将 B 装置 116 的描述跟 A 族 102 中代表 B 装置 116 的翻译模块进行比较。总之，服务器 124 有多个查阅服务可用：每一对（X，Y）一个，其中 X 和 Y 是服务器 124 支持的标准。为了支持一族标准 X 和另一族标准 Y 之间的双向桥接，需要两个查阅服务：（X，Y）和（Y，X）。为了支持具有不同标准 P、Q 和 R 的三个族之间的双向桥接，需要六个查阅服务：（P，Q）、（Q，P）、（P，R）、（R，P）、（Q，R）和（R，Q）。当然，服务器 124 也可以只支持单向桥接。

象装置 104～108 和 112～116 这样的装置常常是复合体。例如，电视机通常都有显示屏、放大器和调谐器。服务器 124 可以首先尝试将这个复合体作为一个整体翻译成具有相同功能的一个新的复合体。如果没有成功，服务器 124 就可以一对一地翻译。这样就得到一部分但是有用的映射。例如，如果 A 族 102 没有定义调谐器，仍然可以将 B 型电视桥接到 A 族作为监视器一样的显示器/放大器装置。如果在标准 A 和标准 B 中的子部件之间不存在一对一关系，就可以采用一对多或者多对多映射。例如，标准 A 可能将音量控制和均衡器效果作为单独一个部件来考虑，而标准 B 则将它们区分为不同的子部件。在这种情况下，B 族 110 中只有音量控制部件而没有均衡器部件的装置不能够跟 A 族 102 桥接。另一方面，A 族 102 中只有一个放大器部件（均衡器和音量控制）的装置不能通过采用子部件的一对二映射跟网络 B 桥接。在多数一般情形中，A 族 102 的一组子部件跟 B 族 110 中的另一组子部件

具有多对多映射。

下一步假设已经发现匹配翻译模块 128，将它下载到桥中去，安装在平台 120 上，按照标准 A 的协议注册。这样一来，A 族 102 的其它应用程序和装置就能够通过模块 128 发现和使用装置 116。模块 128 的安装和注册可以推迟到它在桥 118 的执行环境中执行以后进行。

下面将参考图 2 和图 3 用一个实例说明 HAVi 和通用即插即用（UPnP）家庭网如何桥接。家庭联网领域中软件体系结构的 HAVi、家庭 API 和 Jini 标准已经在上面提到的美国序列号 09/340272（代理人文档号 PHA 23634）中进行了详细讨论，在这里将它引入作为参考。在 HAVi 中，DCM（装置控制模块）是一个软件部件，它代表 HAVi 网络中的装置或者功能。DCM 将 HAVi 定义的 API 暴露给这个装置。DCM 在本质上是动态的：如果一个装置被插入网络或者从网络中拆走，就需要在这个网络中相应地安装或者删除这个装置的 DCM。DCM 是 HAVi 概念的核心，也是支持新装置和功能进入 HAVi 网络的灵活性源泉。

通用即插即用（UPnP）是一种开放式的网络体系结构，它被设计成允许简单，特别是多个销售商的分布式装置和软件应用程序之间的通信。UPnP 的目的在于控制家庭用具，包括家庭自动化、音频/视频、打印机、智能电话等等。UPnP 区分控制点（CP）和被控制装置（CD）。CP 包括例如在 PC 上运行的浏览器、允许用户访问被控制装置提供的功能的无线鼠标垫等等。

UPnP 定义一些协议，用来通过 CP 发现和控制装置。UPnP 不定义一个流水机制，供音频/视频装置使用。一些发现和控制协议是 UPnP 规范的一部分，而其它则被 IETF（因特网工程特别工作组）单独标准化。CP 和装置之间的相互作用是建立在因特网协议（IP）的基础之上的。但是，UPnP 允许非 IP 装置被 IP 装置上运行的软件部件代理。这样一个部件，叫做被控制装置（CD）代理，负责将 UPnP 相互作用翻译给被代理装置。

UPnP 装置具有最低级服务子装置的一个分层结构。装置和服务都有标准类型。装置类型定义它允许包括的子装置或者服务。服务类型定义允许服务包括的行动和状态变量。状态变量说明装置状态，可以用 CP 激活行动来改变状态。状态变量和行动的描述叫做 SCP（服务控制协议）。UPnP 装置用 XML 文件描述它自己。这种文件包括它支持的

服务类型的信息。装置也可以有一个表示服务器，用来让 CP 直接进行 UI 控制。

UPnP 依赖于 AutoIP，没有 DHCP 服务器的时候，它为 IP 装置提供一种手段来获得唯一的地址。在 UDP 多播的基础之上，UPnP 定义一种发现协议，叫做 SSDP（简单服务发现协议）。SSDP 的基础是装置周期性地广播它们提供的服务。广播通知里包括服务行动要发送过去的一个 URL：控制服务器。此外，CP 可以向 UPnP 网络查询特定装置或者服务类型或者实例。

UPnP 依赖于 GENA（一般事件通知体系结构）来定义状态变量下标，在 TCP 的基础之上改变通知机制。

CP 检测到它要使用（通过 SSDP）的一项服务以后，它通过发送 SCP 行动给控制服务器 URL 或者查询状态变量来控制这项服务。行动是用 HTTP POST 消息发送的。这些消息的内容由 SOAP（简单对象访问协议）标准来定义。SOAP 定义在 XML 基础之上的一种远程程序调用机制。

UPnP 中的 HAVi 装置的翻译模块或者软件表示被叫做被控制装置（CD）代理，而 HAVi 中 UPnP 装置的软件表示则被叫做装置控制模块（DCM）。

HAVi 到 UPnP 的桥接

图 2 是家庭网系统 200 的一个框图，它说明从 HAVi 到 UPnP 的桥接，它还用粗箭头说明将在这里是一个数码相机的 HAVi 装置桥接到 UPnP 的一系列步骤。

系统 200 拥有装置 202、204 和 206 形成的一个 UPnP 族。装置 202 包括一盏灯，装置 204 包括一个 MP3 播放机，装置 206 包括一个打印机。系统 200 拥有 TV 208 和数字录像机 210 形成的一个 HAVi 网络族。这些族通过桥 118 连接。

在步骤 212 里，将一个 HAVi 照相机 214 插入 HAVi 的 1394 网络，从而使照相机 214 成为一个活动的 HAVi 节点。

在步骤 216 中，平台部件群 218 上的 HAVi 事件管理器发现增加了这个照相机 214。这个 HAVi 平台收听 HAVi NewSoftwareElement 事件消息，或者收听 HAVi NetworkReset 事件消息，来发现新装置照相机 214。

在步骤 220 中，从平台 218 上的 HAVi 注册上提取照相机 214 的

DCM 的注册属性和它的 FCM 部件，用桥服务器 222 理解的某种格式编码，例如用 XML 编码，并且利用 HTTP POST 发送给桥服务器 222。桥 118 可以使用 HAVi Webproxy FCM 来实现它。

在步骤 224 里，查阅部件用 DCM/FCM 注册属性的形式将 HAVi 装置描述映射到这个装置的 UPnP CD 代理 226。由于 UPnP CD 代理和 HAVi DCM 都是复合体，因此如上所述，位置处理可以在子装置（部件）层次上进行。在 HAVi 中，装置（软件表示是 DCM）包括多个功能部件（软件表示是 FCM)。为了找出哪些 FCM 是 DCM 的一部分，桥 118 可以使用 DCM::GetFcmSeidList 和 FCM::GetFcmType 方法，或者查找 GUID 和 TargetId 属性的 n1 字段具有相同值的注册项。在 UPnP 中，装置是具有最低层服务的子装置的分层结构。FCM 跟服务的目的相同。HAVi 装置到 UPnP CD 代理 226 的映射可以是从整个 DCM 到整个 CD 代理，或者是部分地从 FCM 到代理服务。FCM 到服务的映射可以是一对一、一对多或者多对多的。

在步骤 228 中，在桥 118 的执行环境中运行下载 CD 代理 226。这样做涉及为 CD 代理 226 唯一的 URL 安装 http 服务器。

在步骤 230 中，CD 代理 226 发出周期性的通知消息，并且对发现消息做出响应。这样就使得其它 UPnP 应用和装置能够通过 CD 代理 226 发现和使用 HAVi 照相机 214。

UPnP 到 HAVi 的桥接

图 3 说明将 UPnP 装置，在这里是打印机 206，跟系统 200 中 HAVi 族 208、210、214 进行桥接的步骤。

在步骤 302 中，将 UPnP 打印机 206 插入 UPnP 网络，并且打开这个 UPnP 装置的电源。

下一步 304 是收听 UPnP 装置通知消息，并且对此做出反应。

在步骤 306 中，从嵌入通知消息里的 URL 提取打印机 206 的装置描述文件，用 HTTP POST 将这个文件发送给桥服务器 222。

在步骤 308 中，查找部件用描述文件（XML）形式的 UPnP 装置描述映射成这个装置，在这里是打印机 206，的 HAVi DCM。由于 UPnP CD 代理和 HAVi DCM 是复合体，因此如前所述，定位过程可以在网络装置（部件）一级上进行。在 HAVi 中，（软件表示是 DCM）的装置包括多个功能部件（软件代理是 FCM)。在 UPnP 中，装置是具有最低层服务

的子装置的分层结构。是UPnP装置一部分的服务可以从装置描述文件中找到。FCM跟服务的目的相同。将HAVi装置映射到UPnP CD代理可以是从完整的DCM向完整的CD代理，或者部分地从FCM向服务进行。FCM到服务的映射可以是一对一、一对多或者多对多的。

在步骤310中，在桥118的执行环境中运行下载下来的打印机DCM 312。这样做需要调用DCM的安装方法。

在步骤314中，DCM 312和它的FCM产生HAVi软件单元，并且利用它们在HAVi注册部件那里注册（它是桥118上可以获得的部分平台部件218）。

在步骤316中，HAVi注册为DCM 312和属于它一部分的所有FCM发出一个全局NewSoftwareElement事件消息。这样就使得其它HAVi应用和装置能够通过打印机DCM 312发现和使用UPnP打印机206。

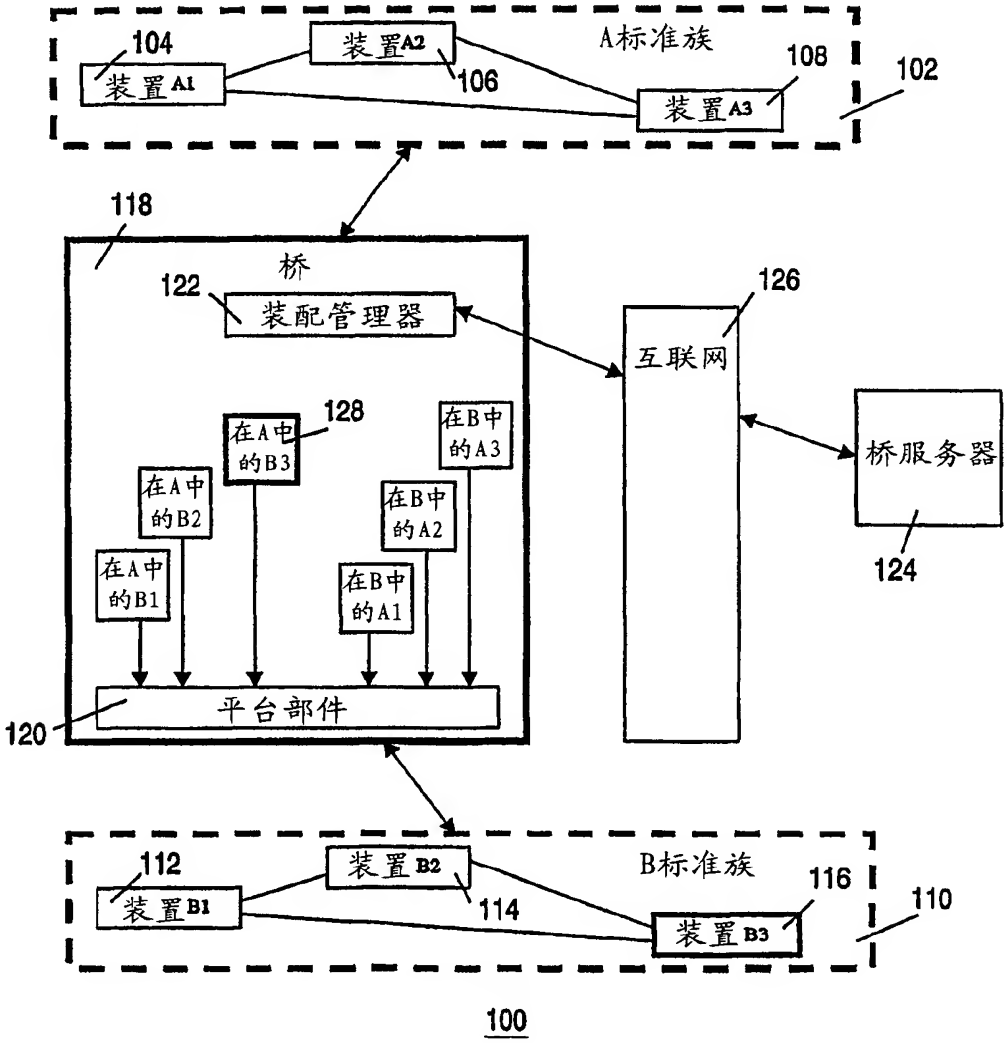

A标准族
104
装置A1
装置A2
106
108
装置A3
102
118
桥
122
装配管理器
126
互联网
桥服务器
124
128
在A中的B3
在B中的A3
在A中的B2
在B中的A2
在A中的B1
在B中的A1
平台部件
120
B标准族
112
装置B1
装置B2
114
116
装置B3
110
100

图 1

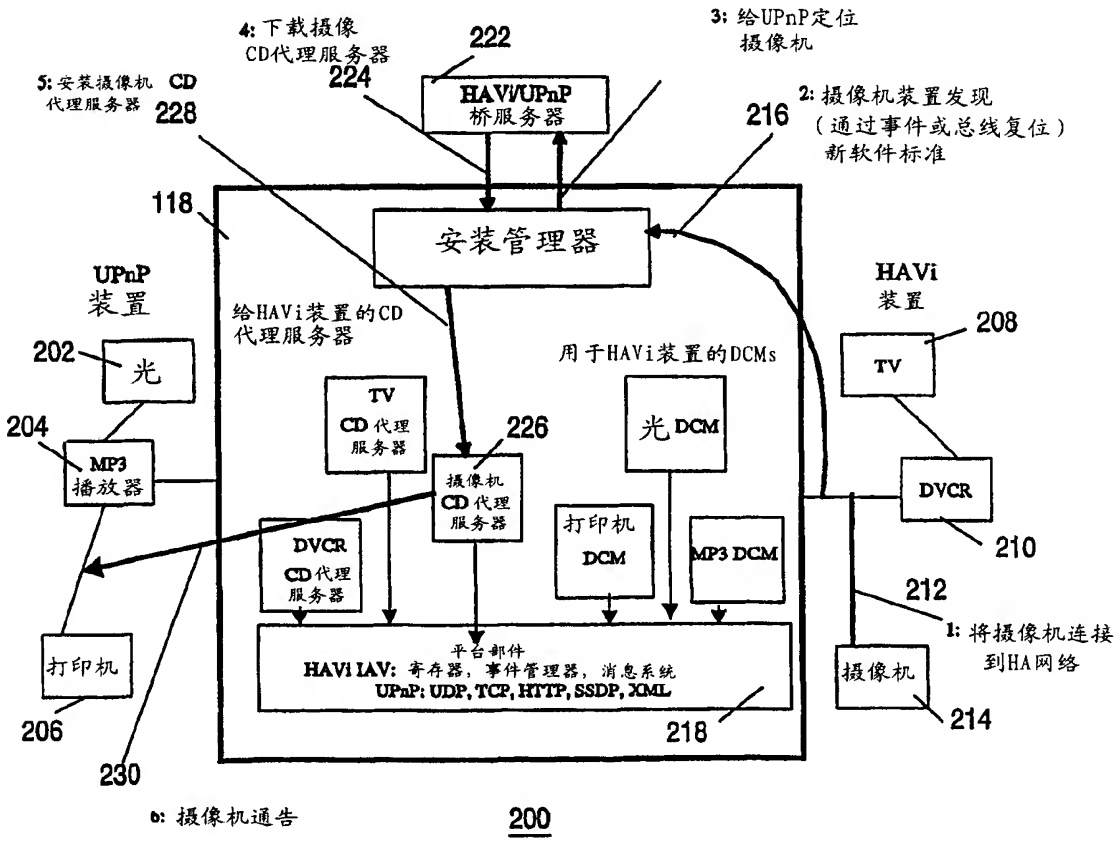

4: 下载摄像
CD代理服务器
222
224
3: 给UPnP定位
摄像机
5: 安装摄像机 CD
代理服务器
228
HAVi/UPnP
桥服务器
216
2: 摄像机装置发现
（通过事件或总线复位）
新软件标准
118
安装管理器
UPnP
装置
给HAVi装置的CD
代理服务器
用于HAVi装置的DCMs
HAVi
装置
208
202
光
TV
CD 代理
服务器
226
光 DCM
TV
204
MP3
播放器
摄像机
CD 代理
服务器
DVCR
210
DVCR
CD 代理
服务器
打印机
DCM
MP3 DCM
212
1: 将摄像机连接
到HA网络
打印机
206
平台部件
HAVi IAV: 寄存器，事件管理器，消息系统
UPnP: UDP, TCP, HTTP, SSDP, XML
摄像机
214
218
230
6: 摄像机通告
200

图 2

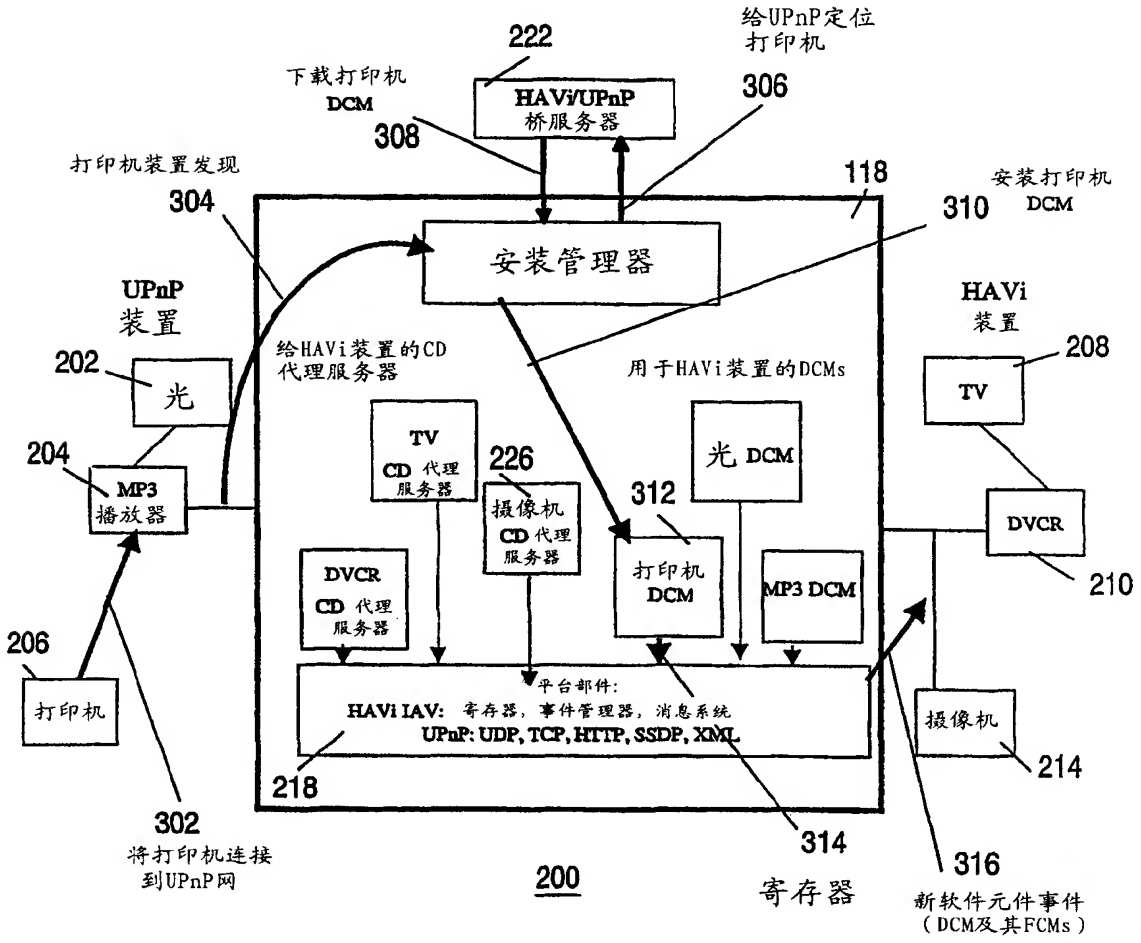

222
给UPnP定位
打印机
下载打印机
DCM
HAVi/UPnP
桥服务器
306
308
打印机装置发现
304
118
安装打印机
DCM
310
安装管理器
UPnP
装置
HAVi
装置
202
光
给HAVi装置的CD
代理服务器
用于HAVi装置的DCMs
208
TV
TV
CD 代理
服务器
226
光 DCM
204
MP3
播放器
312
摄像机
CD 代理
服务器
DVCR
210
DVCR
CD 代理
服务器
打印机
DCM
MP3 DCM
206
平台部件:
HAVi IAV: 寄存器，事件管理器，消息系统
UPnP: UDP, TCP, HTTP, SSDP, XML
摄像机
214
218
302
将打印机连接
到UPnP网
314
316
200
寄存器
新软件元件事件
(DCM及其FCMs)

图 3

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
31 January 2002 (31.01.2002)

PCT

(10) International Publication Number
WO 02/09350 A3

(51) International Patent Classification[7]: H04L 12/12, 12/64, 12/46

(21) International Application Number: PCT/EP01/08552

(22) International Filing Date: 20 July 2001 (20.07.2001)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09/616,632 26 July 2000 (26.07.2000) US

(71) Applicant: KONINKLIJKE PHILIPS ELECTRONICS N.V. [NL/NL]; Groenewoudseweg 1, NL-5621 BA Eindhoven (NL).

(72) Inventors: MOONEN, Jan, R.; Prof. Holstlaan 6, NL-5656 AA Eindhoven (NL). SHTEYN, Yevgeniy, E.; Prof. Holstlaan 6, NL-5656 AA Eindhoven (NL).

(74) Agent: GROENENDAAL, Antonius, W., M.; Internationaal Octrooibureau B.V., Prof. Holstlaan 6, NL-5656 AA Eindhoven (NL).

(81) Designated States *(national)*: CN, JP, KR.

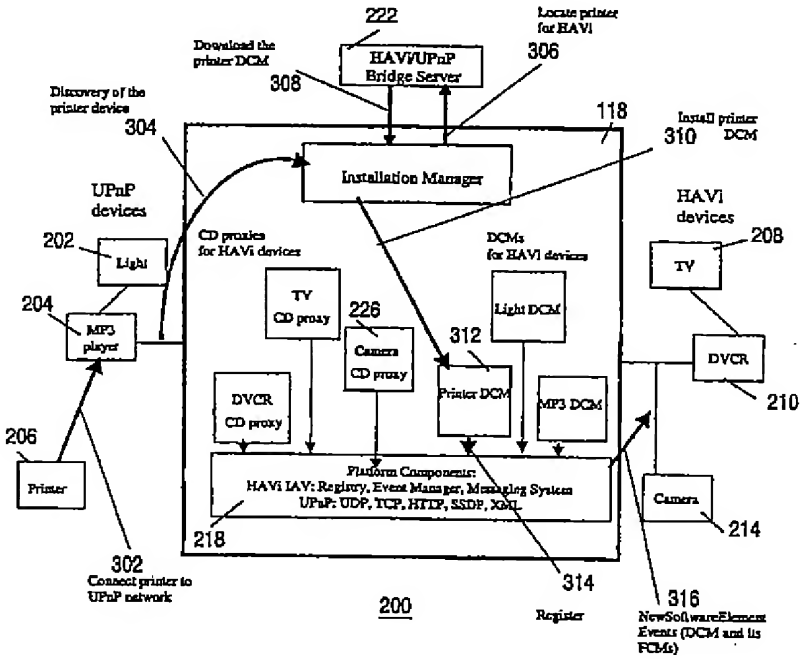
(84) Designated States *(regional)*: European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR).

Published:
— *with international search report*
— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments*

(88) Date of publication of the international search report:
11 April 2002

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) Title: SERVER-BASED MULTI-STANDARD HOME NETWORK BRIDGING



(57) Abstract: A bridge in a home network couples first and second clusters of devices. The clusters have different software architectures. The bridge is connected to a server on the Internet. This server offers a lookup service for some set of standards, and allows a bridge to locate and download the appropriate translation modules for allowing a device in the first cluster to interact with the second cluster.

WO 02/09350 A3

Server-based multi-standard home network bridging

This application is a Continuation-in-Part of U.S. serial no. 09/340,272 (attorney docket PHA 23,634) filed 6/25/99 for Yevgeniy Eugene Shteyn for BRIDGING MULTIPLE HOME NETWORK SOFTWARE ARCHITECTURES.

The invention relates to the bridging of multiple networks based on different software architectures. The invention relates in particular to home networking.

It seems to be unlikely that there will ever be a single universally applicable networking standard for each device in the home. Multiple standards for software architectures coexist and new ones will emerge. New standard interfaces will be developed for new types of devices that are specifically targeted by those standards. Home applications are designed to make an intelligent use of all devices available in the home, but are not capable of dealing with each and every networking standard in use or becoming available in the future. Similarly, devices themselves will not be able to support every existing home networking standard. For these reasons, bridges are needed between the different sub-networks or clusters, each respective one complying with a specific respective standard. A bridge serves to transparently represent a device, complying with a first standard in a network cluster of a first type, as a device complying with a second standard in a network cluster of a second type. The result is a single unified view of the home network available to software applications written for the first standard and as well those written for the second standard.

Typically, a device on a network is controlled through a set of messages complying with the interface of the device. Interoperability between devices and software applications depends upon standard interfaces with a unique identification. Once an application knows the unique identification of the device, it knows the interface of the device and it can control the device by sending it messages. To the application it makes no difference whether it is sending standard messages directly to the device itself, or indirectly via a software component that translates these messages into a different set of message that (eventually) achieve the same desired effect or state change at the controlled device. Hence, a bridge between networks of multiple different standards can be considered a multi-standard device. That is, the bridge complies with each of these multiple standards, and hosts software components that translate device interfaces from a first to a second standard and vice versa.

For example, a bridge is used between a cluster of five devices complying with a standard A and three devices complying with a standard B, different from A. The bridge hosts three translation modules for translating from A to B and five translation modules for translating from B to A. Accordingly, a software application capable of interacting with only standard A devices can now control all eight devices.

U.S. serial no. 09/340,272 (attorney docket PHA 23,634) filed 6/25/99 for Yevgeniy Eugene Shteyn for BRIDGING MULTIPLE HOME NETWORK SOFTWARE ARCHITECTURES, herein incorporated by reference, relates to the bridging of home networks of different software architectures. References to software representations of devices and services on a first one of the networks are automatically created. The references are semantically sufficient to enable automatic creation of at least partly functionally equivalent software representations for a second one of the networks so as to make the devices and services of the first network accessible from the second network. This document also addresses the HAVi, Home API and Jini software architectures.

In the following, the expression "translation module" includes the concept of "software representation", i.e., the representation in software of a physical device or of a service on a network or a sub-set thereof so as to make the device or service accessible to the relevant messaging or controlling software.

Preferably, a bridge performs the following functions: detection of the addition of a device in either of the bridged networks; identification of the type of the added device; locating the translation module for the identified device type if the device is likely to be of interest to the other network; and installing the translation module on the other network according to the procedure required by the standard used by that network.

Successful standards will continue to define standard interfaces for newly developed devices in a domain considered relevant to those standards. This necessitates the development of accompanying translation modules that enable those new devices to be represented in networks based on different standards. As a consequence, a bridge is unable to contain all embedded translation modules for all possible, relevant devices that become available in the future.

Accordingly, the inventors propose a solution wherein a bridge is connected to a server, e.g., on the Internet. This server offers a lookup service for some set of standards, and allows a bridge to locate and download the appropriate translation modules for use in the home network.

More specifically, the invention relates to a method of providing a service to a user of a home network. The method comprises enabling a component of a first cluster in the network to interact with a second cluster of the home network. The first cluster has a first software architecture, and the second cluster has a second software architectures different from the first one. The first and second clusters are coupled through a bridge. The method comprises enabling a server external to the clusters, e.g., on the Internet, to receive a reference of a component having a first software representation in the first cluster. The bridge may provide this reference. The method further comprises enabling to provide to the bridge a translation module, associated with the reference, for at least partially representing the component on the second cluster upon the module being installed on the bridge.

A service provider thus can maintain and update a data base of translation modules for any multiple standards being used in home networks. This partitioning and delegation of functionalities has many advantages as discussed below.

The invention allows the bridge to be fairly "light-weight" or low-cost, as it does not need to have embedded translation modules for all possible devices of all standards that it could be connected to in the home. Storage and computation power is only needed for the devices that are actually bridged (i.e., storage is not needed for the ones that are *potentially* bridged), and only *when* they are bridged (e.g., "just-in-time"-bridging, i.e., at the time of connecting the new device to the network).

Further, the invention renders the home network as a whole extensible and future-proof. As new devices are invented and descriptions thereof become part of various standard specs, these descriptions are translated and uploaded to the bridge server by, e.g., the device manufacturer or a third party, to make them available for bridging in existing home networks. This process does not require any update mechanism of components in the home network itself.

A further benefit is that the bridge-server operator is able to obtain information about the configuration of the individual user's home network. This information can be used to the advance of both the user and the manufacturers and service providers. See, for example, U.S. Serial No. 09/160,490 (attorney docket PHA 23,500) filed 9/25/1998 for Adrian Turner et al., for CUSTOMIZED UPGRADING OF INTERNET-ENABLED DEVICES BASED ON USER-PROFILE, incorporated herein by reference. This document relates to a server system that maintains a user profile of a particular end-user of consumer electronics network-enabled equipment and a data base of new technical features for this type of equipment, e.g., a home network. If there is a match between the user-profile and a new

technical feature, and the user indicates to receive information about updates or sales offers, the user gets notified via the network of the option to obtain the feature. Also see, e.g., U.S Serial No.09/189,535 (attorney docket PHA 23,527) filed 11/10/98 for Yevgeniy Shteyn for UPGRADING OF SYNERGETIC ASPECTS OF HOME NETWORKS, incorporated herein by reference. This document relates to a system with a server that has access to an inventory of devices and capabilities on a user's home network. The inventory is, for example, a look-up service as provided by HAVi, Jini and Home API architectures. The server has also access to a data base with information of features for a network. The server determines if the synergy of the apparatus present on the user's network can be enhanced based on the listing of the inventory and on the user's profile. If there are features that are relevant to the synergy, based on these criteria, the user gets notified. In this sense, U.S. serial no. 09/189,535 relates to the concept of an "application suggestor".

A further advantage of the invention resides in the fact that the server operator is enabled to measure market demands for particular devices to be bridged to a specific standard. The device manufacturer or another relevant third party can be notified of the emerging demand. When the new translation modules become available on the server, bridges that have sent requests for translation modules in the past with which the server could not comply, can now be notified of an upgrade.

Note that the bridge can be implemented as a software component on a device of a specific cluster on the home network in order to provide a bridge to another cluster. For example, a HAVi set top box can have a software component for bridging the HAVi cluster to, e.g., a UPnP cluster on the network. Similarly, a PC that controls the UPnP cluster can have a software component that serves to bridge the UPnP cluster of the home network to the HAVi cluster.

The invention is explained in further detail, and by way of example, with reference to the accompanying drawing, wherein:

Fig.1 is a block diagram illustrating the principle of the bridging between two networks according to the invention;

Fig.2 is a block diagram illustrating bridging HAVi to UPnP; and

Fig.3 is a block diagram illustrating bridging UPnP to HAVi.

Throughout the drawing, same reference numerals indicate similar or corresponding features.

As mentioned above an aspect of the invention relates to connecting a bridge to a server, e.g., on the Internet. This server offers a lookup service for some set of standards, and allows a bridge to locate and download the appropriate translation modules in the home network that eventually enables a device on a sub-network of a first architecture to work with devices on a sub-network of a second architecture.

Fig.1 is a diagram of a home network system 100 with a first cluster 102 of devices 104, 106 and 108 that comply with a first software architecture standard, herein-after called standard A. System 100 comprises a second cluster 110 of devices 112, 114 and 116 that comply with a second software architecture standard, herein-after called standard B. Clusters 102 and 110 are interconnected through a bridge 118. In order to have a meaningful network interaction between cluster 102 of standard A on the one hand, and cluster 110 of standard B on the other hand, translation modules are introduced. These modules need to participate in both clusters 102 and 110. The modules typically need components local to the clusters, such as lower-level communication software, in order to be capable of this participating. Rather than having each translation module comprising its own communication software it is more efficient to provide this software as an element of platform components 120 of bridge 118.

The process of the invention is now illustrated with an example wherein B-device 116 is going to be added to system 100.

The first step comprises physically connecting B-device 116 to B-cluster 110, or "booting" B-device 116.

In a next step, bridge 118 detects B-device 116 as a new addition, either because bridge 118 scans B-cluster 110 or its registry/directory/look-up service (not shown) periodically or because B-cluster 110 actively notifies bridge 118. Bridge 118 comprises a software component 122, referred to as Installation Manager, that handles the installation of further software components needed to integrate B-device 116 into system 100. An aspect hereof is discussed in, e.g., U.S. serial no. 09/340,272 (attorney docket PHA 23,634). In the latter document a software component, referred to as the Reference Factory, is capable of extracting information from any of the software representations of devices registered. This Reference Factory is capable of querying the inventory of services or of getting notified of a new software representation according to the methods of the relevant software architecture. Similarly, Installation Manager 122 receives or retrieves information descriptive of newly

added B-device 116. The descriptive information is possibly reformatted before being sent to a bridge server 124 via the Internet 126. In addition, bridge 118 preferably provides information about the local execution environment of home network 100. This information is relevant to the software components that server 124 downloads onto bridge 118. The relevant information regarding the environment relates to the software architectures present, in this case the A-standard cluster 102 and the B-standard cluster 110. The information may also relate to the memory available, the type of operating system(s) being used, virtual machines present, platform libraries, etc., on bridge 118. Based on this information server 124 is able to select the proper translation module or modules that fits or fit in best with the network environment of system 100.

Upon receipt of the descriptive and environment information, server 124 uses a look-up service that needs to match the information, descriptive of B-device 116, with a translation module for representing B-device 116 in A-cluster 102. In general, server 124 has a plurality of look-up services available: one for each ordered pair (X,Y), wherein X and Y are the standards supported by server 124. In order to support bi-directional bridging between a cluster of standard X and another cluster of standard Y, two look-up services are needed: (X,Y) and (Y,X). For support of bi-directional bridging between three clusters, all with different standards P, Q and R, six look-up services are needed: (P,Q); (Q,P); (P,R); (R,P); (Q,R); and (R,Q). Of course, server 124 may only support uni-directional bridging.

Devices, such as devices 104-108 and 112-116, are often composite objects. For example, a TV set typically comprises Display, Amplifier and Tuner components. Server 124 could first try to translate the composite object as a whole into a new composite device with an equivalent functionality. If that does not succeed server 124 could translate as much subsidiary components as possible on a one-to-one basis. This then would result in a partial, but still useful, mapping. For example, if A-cluster 102 has not available a definition of a Tuner, a B-type TV could still be bridged to A-cluster 102 as a monitor-like display/amplifier device. If a one-to-one relationship does not exist between particular subsidiary components in standard A and standard B, then a one-to-many or many-to-many mapping could be used. For example, standard A might define volume control and equalizer effects as a single subsidiary component, whereas standard B distinguishes them as separate subsidiary components. In this case, a device in B-cluster 110 that contains just the volume control component, but not the equalizer component, cannot be bridged to A-cluster 102. On the other hand, a device in A-cluster 102 that contains a single amplifier component (volume control as well as equalizer) can be bridged to network B by applying a one-to-two mapping

of the subsidiary components. In the most general case, a particular set of subsidiary components in A-cluster 102 matches with another set of subsidiary components in B-cluster 110 under a many-to-many mapping.

Next, assume that a matching translation module 128 has been found it is downloaded to the bridge, installed on platform 120 and registered in accordance with the protocol of standard A. This enables other applications and devices of A-cluster 102 to discover and use device 116 through module 128. The installation and registering of module 128 may be postponed until after it has been run on the execution environment of bridge 118.

The invention is explained below with an example illustrating the bridging of HAVi and Universal Plug and Play (UPnP) home networks with reference to Figs.2 and 3. The HAVi, Home API, and Jini standards for software architectures in the home networking field have been discussed to some detail in U.S. serial no. 09/340,272 (attorney docket PHA 23,634) mentioned above and incorporated herein by reference. In HAVi, a DCM (Device Control Module) is a software element that represents a single device or functionality on the HAVi network. The DCM exposes the HAVi defined APIs for that device. DCMs are dynamic in nature: if a device is inserted or removed from the network, a DCM for that device needs to be installed or removed, respectively, in the network. DCMs are central to the HAVi concept and the source of flexibility in accommodating new devices and features into the HAVi network.

Universal Plug and Play (UPnP) is an open network architecture that is designed to enable simple, ad hoc communication among distributed devices and software applications from multiple vendors. UPnP leverages Internet technology and extends it for use in non-supervised home networks. UPnP aims at controlling home appliances, including home automation, audio/video, printers, smart phones, etc. UPnP distinguishes between Control Points (CPs) and controlled devices (CDs). CPs comprise, e.g., browsers running on PCs, wireless pads, etc., that enable a user to access the functionality provided by controlled devices.

UPnP defines protocols for discovery and control of devices by CPs. UPnP does not define a streaming mechanism for use by Audio/Video devices. Some of the discovery and control protocols are part of the UPnP specification while others are separately standardized by the IETF (Internet Engineering Taks Force). Interaction between CPs and devices is based on the Internet protocol (IP). However, UPnP allows non-IP devices to be proxied by a software component running on IP-compliant devices. Such a component, called

Controlled Device (CD) proxy, is responsible for translation and forwarding of UPnP interactions to the proxied device.

A UPnP device has a hierarchy of sub-devices with at the lowest level services. Both devices and services have standardized types. A device type determines the sub-devices or services that it is allowed to contain. A service type defines the actions and state variables that a service is allowed to contain. State variables model the state of the device, actions can be invoked by a CP in order to change that state. The description of the state variables and the action is called the SCP (Service Control Protocol). A UPnP device provides a description of itself in the form of an XML document. This document contains, among other things, the service types that it supports. Optionally, a device may have a presentation server for direct UI control by a CP.

UPnP relies on AutoIP, which provides a means for an IP device to get a unique address in the absence of a DHCP server. UPnP defines a discovery protocol, based on UDP multicast, called SSDP (Simple Service Discovery Protocol). SSDP is based on devices periodically multicasting announcements of the services that they provide. An announcement contains a URL to which service actions are to be sent: the control server. In addition to that, CPs may query the UPnP network for particular device or services types or instances.

UPnP relies on GENA (Generic Event Notification Architecture) to define a state variable subscription and change notification mechanism based on TCP.

After a CP has detected a service it wants to use (via SSDP), it controls the service by sending SCP actions to the control server URL or querying for state variables. Actions are sent using HTTP POST messages. The body of these messages are defined by the SOAP (Simple Object Access Protocol) standard. SOAP defines a remote procedure call mechanism based on XML.

The translation modules, or software representations, of HAVi devices in UPnP are called Controlled Device (CD) *proxies*, while the software representations of UPnP devices in HAVi are called *Device Control Modules* (DCMs).

HAVi to UPnP bridging

Fig 2 is a block diagram of a home network system 200 illustrating the bridging from HAVi to UPnP, and shows in bold arrows the sequence of steps to bridge a HAVi device, here a digital camera, to UPnP.

System 200 has a UPnP cluster formed by devices 202, 204, and 206. Device 202 comprises a light, device 204 comprises an MP3 player, and device 206 comprises a

printer. System 200 has a HAVi network cluster formed by a TV 208, and a digital video recorder 210. The clusters are connected through bridge 118.

In a step 212 a HAVi Camera 214 is physically plugged into the HAVi's 1394 network, thus making Camera device 214 an active HAVi node.

In a step 216 this addition is discovered by the HAVi Event Manager, residing on platform components group 218. The HAVi platform listens and reacts to the HAVi NewSoftwareElement event, or listens to a HAVi NetworkReset event to discover Camera 214 as new device.

In a step 220 the registration attributes of the DCM of Camera 214 and its FCM components are retrieved from the HAVi Registry on platform 218, and are encoded in some format understood by a bridge server 222, for example XML, and are sent to bridge server 222 using HTTP POST. Bridge 118 may use a HAVi Webproxy FCM to implement this.

In a step 224 a look-up component maps a HAVi device description, in the form of DCM/FCM registration attributes, to a UPnP CD proxy 226 for that device. Since UPnP CD proxies and HAVi DCMs are composite objects, the location process might be implemented on the sub device (component) level, as described earlier. In HAVi a device (software representation is DCM) consists of a number of *functional components* (software representation is FCM). To find out which FCMs are part of a DCM bridge 118 can use the DCM::GetFcmSeidList and FCM::GetFcmType methods, or look at registry entries that have the same values for the GUID and n1 field of the TargetId attribute. In UPnP, a device is a hierarchical structure of sub devices with at the lowest level *services*. FCMs serve the same purpose as services. The mapping of HAVi device to UPnP CD proxy 226 might be from complete DCM to complete CD proxy, or partially from FCMs to proxy services. The mapping of FCM to service might be 1-to-1, 1-to-many or many-to-many.

In a step 228 downloaded CD proxy 226 is run on the execution environment of bridge 118. This involves installing an http server for the unique URL of CD proxy 226.

In a step 230 CD proxy 226 sends out periodic announcement messages, and responds to discover messages. This enables the other UPnP applications and devices to discover and use the HAVi Camera 214 through the CD proxy 226.

UPnP to HAVi bridging

Fig.3 illustrates the steps to bridge a UPnP device, here printer 206 to HAVi cluster 208, 210, 214 in system 200.

In a step 302, UPnP Printer 206 is physically plugged into the UPnP network, and 'powering-up' the UPnP device.

A next step 304 involves listening and reacting on the UPnP device announcement message.

In a step 306, the device description document of printer 206 is retrieved from the URL embodied in the announcement message, and the document is sent to bridge server 222 using HTTP POST.

A step 308 involves a look-up component that maps a UPnP device description, in the form of a description document (in XML), to a HAVi DCM for that device, here printer 206. Since UPnP CD proxies and HAVi DCMs are composite objects, the location process might be implemented on the sub device (component) level, as described earlier. In HAVi a device (software representation is DCM) consists of a number of *functional components* (software representation is FCM). In UPnP, a device is a hierarchical structure of sub devices with, at the lowest level, *services*. Services that are part of a UPnP device can be found in the device description document. FCMs serve the same purpose as services. The mapping of HAVi device to UPnP CD proxy might be from complete DCM to complete CD proxy, or partially from FCMs to services. The mapping of FCM to service might be 1-to-1, 1-to-many or many-to-many.

A step 310 involves running a downloaded Printer DCM 312 in the execution environment of bridge 118. This involves calling the DCM's Install method.

A step 314 involves DCM 312 and its FCMs to create HAVi software elements, and using those to register with the HAVi registry component (which is part of platform components 218 available on bridge 118).

In a step 316 the HAVi registry posts global NewSoftwareElement events for DCM 312 and all FCMs that are part of it. This enables the other HAVi applications and devices to discover and use UPnP Printer 206 through Printer DCM 312.

CLAIMS:

1. A method of providing a service to a user of a home network (100), wherein:

- the method comprises enabling a component (116) of a first cluster (110) in the network to interact with a second cluster (102) of the home network;
- the first cluster has a first software architecture;
- the second cluster has a second software architectures different from the first architecture;
- the first and second clusters are coupled through a bridge (118);
- the method comprises:

- enabling a server (124) external to the clusters to receive a reference of a component having a first software representation in the first cluster; and

- enabling to provide to the bridge a translation module (128), associated with the reference, for at least partially representing the component on the second cluster upon the module being installed on the bridge.

2. The method of claim 1, wherein the server receives the reference from the bridge.

3. The method of claim 1, wherein the bridge contacts the server via the Internet.

4. The method of claim 1, wherein the first cluster comprises a HAVi cluster.

5. The method of claim 1, wherein the first cluster comprises a UPnP cluster.

6. A data base comprising at least one translation module (128) for enabling a component (116) on a first home network cluster (110) of a first software architecture to interact with a second home network cluster (102) of a second software architecture upon the module having been downloaded via the Internet on a bridge (118) that couples the first and second clusters.

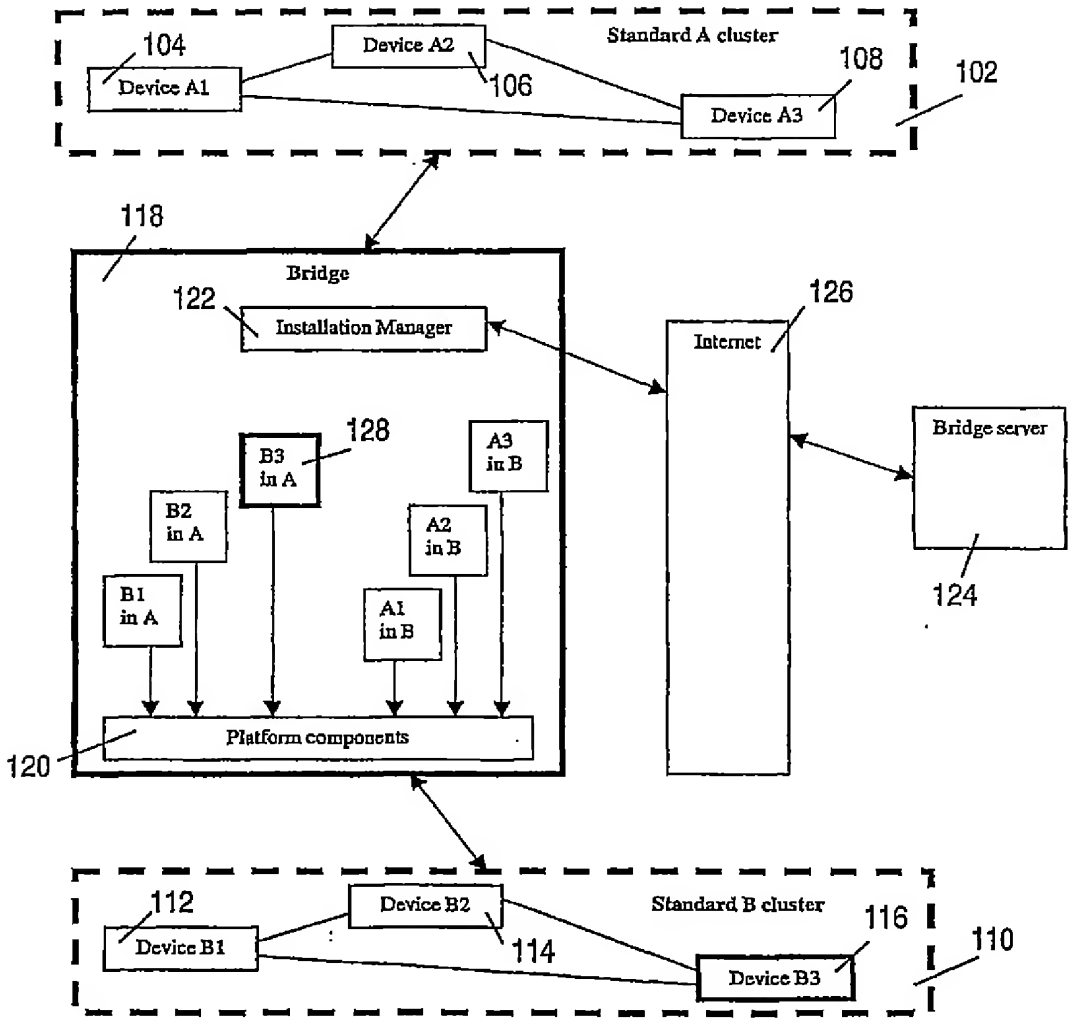

Standard A cluster
104
Device A1
Device A2
106
108
Device A3
102
118
Bridge
122
Installation Manager
126
Internet
Bridge server
124
128
B3 in A
A3 in B
B2 in A
A2 in B
B1 in A
A1 in B
Platform components
120
112
Device B1
Device B2
114
Standard B cluster
116
Device B3
110
100

FIG. 1

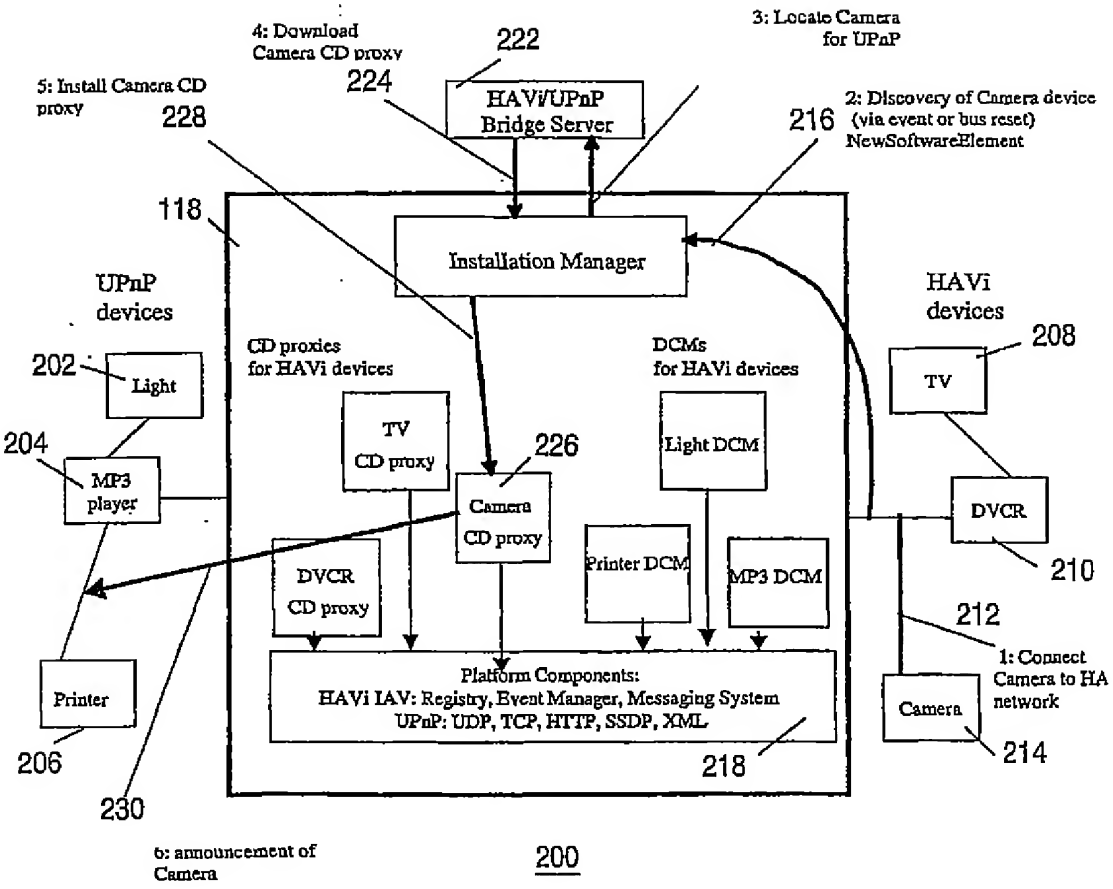

4: Download
Camera CD proxy
222
224
3: Locate Camera
for UPnP
5: Install Camera CD
proxy
228
HAVi/UPnP
Bridge Server
216
2: Discovery of Camera device
(via event or bus reset)
NewSoftwareElement
118
Installation Manager
UPnP
devices
HAVi
devices
CD proxies
for HAVi devices
DCMs
for HAVi devices
208
202
Light
TV
TV
CD proxy
226
Light DCM
204
MP3
player
Camera
CD proxy
DVCR
210
DVCR
CD proxy
Printer DCM
MP3 DCM
212
1: Connect
Camera to HA
network
Platform Components:
HAVi IAV: Registry, Event Manager, Messaging System
UPnP: UDP, TCP, HTTP, SSDP, XML
Printer
Camera
206
218
214
230
6: announcement of
Camera
200

FIG. 2

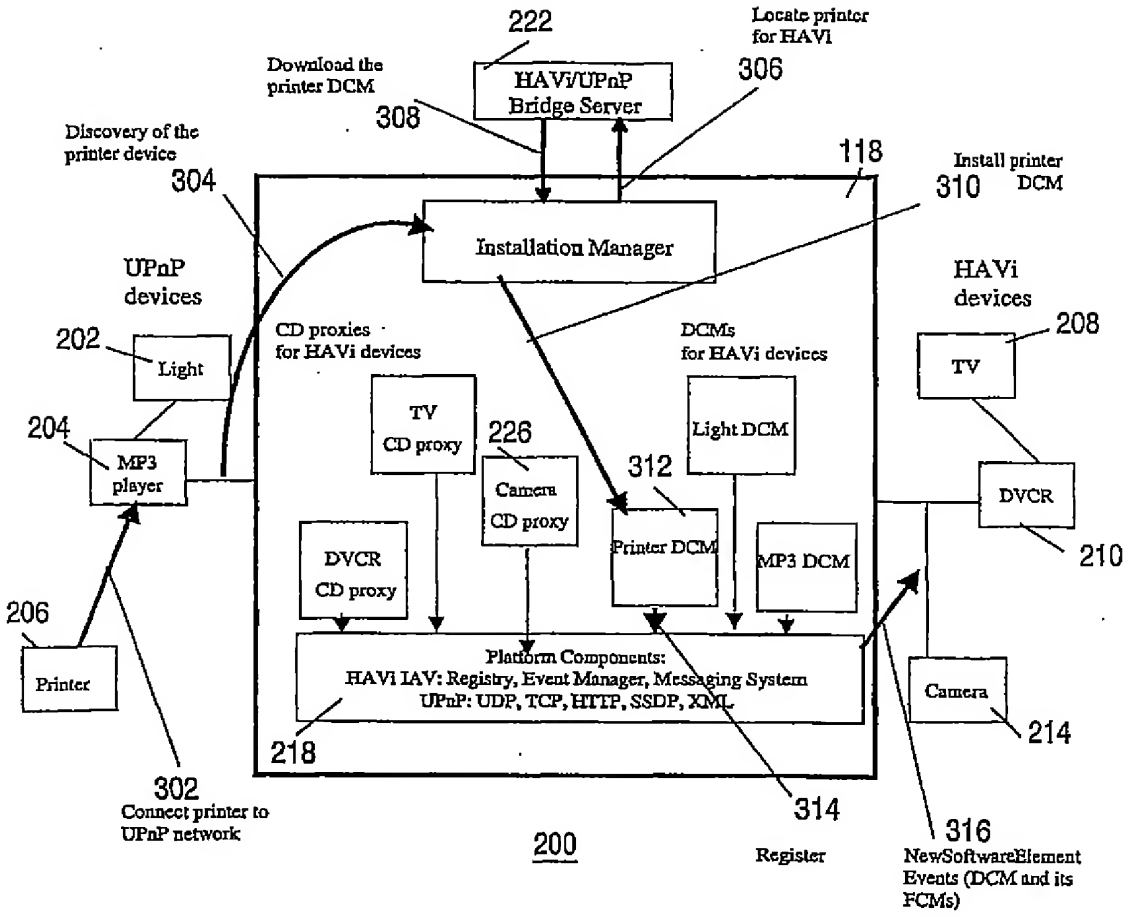

222
Locate printer for HAVi
Download the printer DCM
HAVi/UPnP Bridge Server
306
308
Discovery of the printer device
304
118
Install printer DCM
310
Installation Manager
UPnP devices
HAVi devices
CD proxies for HAVi devices
DCMs for HAVi devices
202
Light
208
TV
204
MP3 player
TV CD proxy
226
Light DCM
Camera CD proxy
312
DVCR
210
DVCR CD proxy
Printer DCM
MP3 DCM
206
Printer
Platform Components:
HAVi IAV: Registry, Event Manager, Messaging System
UPnP: UDP, TCP, HTTP, SSDP, XML
Camera
214
218
302
Connect printer to UPnP network
314
Register
316
NewSoftwareElement Events (DCM and its FCMs)
200

FIG. 3